Overview of the control system for the Keck Interferometer

A. J. Booth*[a], G. Eychaner*[a], E. Hovland*[a], R. L. Johnson Jr.*[a], W. Lupton**[b], A. Niessner*[a], D. Palmer*[a], L. J. Reder*[a], A. C. Rudeen**[b], R. F Smythe*[a], K. Tsubota**[b]
[a]Jet Propulsion Laboratory, California Institute of Technology, [b]W. M. Keck Observatory, California Association for Research in Astronomy

ABSTRACT

The Keck Interferometer links the two 10m Keck Telescopes located atop Mauna Kea in Hawaii. It is the first 10m class, fully AO equipped interferometer to enter operation. Further, it is the first large interferometer designed to be handed over from a design and implementation team to a separate operations team, and be used by astronomers who are not interferometer specialists. As such it offers unique challenges in reducing an extremely complex and powerful system to an apparently simple user interface, and providing a well engineered system that can be maintained by people who did not develop it.

This paper gives an overview of the control system that has been implemented for the single baseline operation of the instrument, and indicates how this will be extended to allow control of the future modes of the instrument (nulling, differential phase and astrometry).

The control system has several parts. One is for control of "slow" sub-systems, which is based in the EPICS architecture, already ubiquitous at the Keck Observatory. Another, used to control hard real time sub-systems, is based on a new infrastructure developed at JPL, programmed in C++, Java, and using CORBA for communication. This infrastructure has been developed specifically with the problems of interferometric control in mind and is used in JPL's flight testbeds as well as the Keck Interferometer. Finally, a user interface and high level control layer is in development using a variety of tools including UML based modeling in the Rhapsody tool (using C++ and CORBA), Java, and Tcl/Tk for prototyping.

Keywords: Interferometry, control system, real time control

1. INTRODUCTION

1.1. Overview

The Keck Interferometric Array, as proposed, would use Michelson combination among the two Keck 10m telescopes along with four proposed 1.8m outrigger telescopes. The two Kecks provide a baseline of 85 m in the NE direction. With the addition of the outriggers, a range of baselines from 30 m to 135 m could be provided. For highest efficiency, all of the telescopes are phased: the Kecks use adaptive optics, while the outriggers would use fast tip/tilt correction. Cophasing among baselines is provided using active fringe tracking and active delay lines; for off-source cophasing, a dual-star module will be installed at each telescope to bring the light from the source and a cophasing reference to the beam combining lab. The back-end science instruments of the interferometer include two-way combiners at 1.5-2.4 μm for astrometry, traditional visibility science, and cophasing; a proposed multi-way imaging combiner at 1.5-5 μm; and a nulling combiner at 10 μm in development. The Keck Interferometer is the first 10m class, fully AO equipped interferometer to enter operation and is described in detail by Colavita and Wizinowich[2,3]. The Keck Interferometer is an element in NASA's Origins Program[1].

* Andrew.J.Both@jpl.nasa.gov; phone 1 818 354 1200; fax 1 818 393 4357; Jet Propulsion Laboratory, MS171-113, 4800 Oak Grove Drive, Pasadena, CA 91109; **ktsubota@keck.hawaii.edu; phone 1 808 885 7887; fax 1 808 885 4464; W. M. Keck Observatory, 65-1120 Mamalahoa Highway, Kamuela, HI 96743

The science goals of the interferometer are the use of nulling interferometry to characterize zodiacal dust emission from nearby solar type stars as a precurser to the SIM and TPF planet finder missions (nulling mode); to detect hot Jupiters (Jovian sized planets in close orbit around their parent stars) using differential phase signals at varying detector wavelengths (DP mode); astrometric searches for planets down to Uranus mass around nearby stars (astrometry mode); and general purpose high resolution astronomical imaging of a variety of sources using the full array (imaging mode). For a detailed description of the science goals of the instrument see Colavita and Wizinowich[2].

1.2. Control system

The control system for the Keck Interferometer is large and complicated. There are many parts and many interfaces, and it is being developed by two physically distant groups – JPL and WMKO, so there is ample potential for development of a system that is difficult to use and maintain. In addition, the system must be handed over from the development group (which for the interferometer itself, as opposed to telescope systems, is dominated by JPL personnel) to the operations group, which will contain none of the JPL developers. Further, the instrument must be of facility class, such that it may be used by astronomers who are not interferometry specialists. Thus, it is requirement that the instrument be, in fact, easy to use and maintain. In addition there are very tight requirements on the performance of the system to reach the challenging science goals of, for example, nulling detection of exo-zodiacal emission and differential phase measurements of hot Jupiters. These requirements are not easy to achieve.

The main components of the control system are:

- **Angle trackers,** which keep the light from the telescopes falling on the entrance aperture to the fringe trackers, and keep the wavefronts that are being combined to form interference fringes parallel.
- **Delay lines,** which keep the optical path difference between any two telescope beams that are being combined the same to within very tight tolerances whilst tracking the moving delay cause by the sidereal motion of the target object.
- **Metrology,** which monitors and corrects for path length and tilt variations in the optical beam trains of the interferometer from the telescopes to the detectors
- **Fringe trackers,** which not only measure the fringe position and provide feed back to delay lines to maintain lock on the interference, but also provide the measurements for phase and visibility that form the science observations.
- **Nulling and imaging cameras,** which will measure the fringe quantities of interest to nulling and imaging science.
- **Telescope systems,** being the telescope and dome control themselves, and also target acquisition, adaptive optics (AO) control, and coudé beam train control (particularly for off axis beam trains used for the secondary star feed).

For most of these sub-systems, several copies of functionally identical, but differently configured versions of the sub-systems are used together. For example, in a full imaging mode deployment there would be twelve delay lines operating simultaneously. There are numerous interfaces between the different sub-systems leading to a complicated data flow in the instrument. For example, as mentioned above, the fringe trackers measure the position of the interference fringes formed by combining the light from two telescopes. This position is fed back to the delay lines so as to keep the center of the interference fringe packet on the beam combiner, to maintain the highest fringe visibility. As another example the angle trackers feed back to the AO systems to keep the target object aligned with the detectors.

The tolerances on the performance of the sub-systems are very demanding. Servos must run at fast rates, for example the delay lines internally servo their positions against a metrology source at 5kHz. The angle trackers and fringe trackers need to read out their camera hardware at close to the maximum allowable rate to provide enough bandwidth for fringe tracking control and angle tracking control to overcome the variability induced by the earth's atmosphere and the internal variations in the instrument (fringe tracker frames at up to 500Hz and angle tracker frames at up to 100Hz). The servo control must also be very precise. The fringe tracking must control the fringe position to much better than a wavelength of light (so to better than 100nm or so), even though the delay lines are moving over 10s of meters during an observation.

To successfully make science measurements with the instrument there must be a high degree of automation for the control system. This also naturally leads to a control system that is easy to use since the user need not have a detailed

knowledge of the internal workings of the subsystems to achieve a set of science observations. The main reason for this automation requirement is that interferometry science requires high levels of careful calibration, thus we require:

- Fast switching between taking science data and taking internal calibration data.
- Fast switching between science target objects and objects of known visibility and phase for external fringe calibration.
- Consistency of calibration processes from targets to calibrator objects and in set up of target and calibrators.

These requirements demand automation at several levels:

- Automated sequencing of sub-system operation within a subsystem.
- Automated sequencing of sub-systems for coordinated operation of subsystems.
- Automated sequencing of targets presented to the instrument.
- Automated optical alignments of system and sub-system optical elements

This last automation requirement arises because the interferometer has many optical elements in each beam train from telescope to detector (many 10s of elements), and the total optical path is very long (many 10s of meters). Thus, manual alignment would be too time consuming and also not of sufficient accuracy.

2. DESIGN APPROACH

2.1. Software

The design approach taken for the interferometer control system was driven by the above complexities and requirements. In this section we give an overview of the thinking behind the current and planned implementations.

At WMKO extensive use is already made of the EPICS[4] control framework. Many observatory sub-systems, including telescope control, are written using EPICS, and there are additional tool sets developed by WMKO which are layered on top of EPICS, for example a keyword interface layer (KTL). There thus exists a large body of expertise and knowledge at WMKO in the use of EPICS and many control systems are built on thoroughly tested device control modules built in EPICS. It was therefore decided wherever possible, to make use of existing EPICS developments at WMKO for control of interferometer sub-systems, and to use WMKO expertise to extend existing control systems where necessary to control interferometer sub-systems that are similar to these existing control systems. This not only builds on existing infrastructure, but also helps the handover process as WMKO personnel are already familiar with the details of control systems for these interferometer elements, even if they were developed by JPL personnel. Care was taken to also make the development and deployment environment of EPICS systems at JPL adhere to WMKO policy and procedures.

Much of the interferometer's control system must run at rates too high to be implemented using EPICS. For these parts of the system we chose to implement control using the JPL developed RTC tool set[5]. The use of this tool set is described more fully below, however, here we note that the tool set allows a modular approach to design of the control system. We have many subsystems that are very similar in concept (for example, angle trackers and fringe trackers both read out a camera at high rate, compute a target, and send the target to another subsystem for actuation); we have many copies of the same type of subsystem, as noted above for example for the 12 delay lines needed in imaging mode. The object oriented nature of the RTC tool set allows easy modularisation of the elements of the control system to achieve this repeatability in design and in instantiation.

In addition to being implemented as a modular system, the control system is also very hierarchical. This particularly applies to the sequencing type of control for the system and for the sub-systems. Sequencing occurs at all levels within in the control system: At the lowest level, the sub-systems sequence their internal states based on low level input from sensors. For example, is there light in the detector, then start to track the fringes. At an intermediate level, sub-systems must be coordinated to act in concert. For example the observational target must be acquired by the angle trackers before the fringe trackers can try to track fringes. Also, the control system must cycle through various states of the sub-systems to achieve an observing sequence with the required internal calibrations. Finally at the highest level the control system must cycle through a predefined list of targets and calibration sources to achieve scientifically interesting measurements. The hierarchical nature of this sequencing allows us to implement it in a modular and hierarchical way, with separate control objects for each level of the hierarchy, linked by common format interfaces. This allows for

separate development and testing, and implementation of prototypes for various levels along side more fully developed systems at other levels.

For sub-systems developed within the EPICS framework, we also employed hierarchical development, layering sequencers on top of collections of individual sub-system modules, to implement, for example, the auto-alignment system.

We made extensive use of prototyping and iterative development of sub-systems and sequencers. Sub-systems are developed as stand alone modules, then interfaced to other subsystems and sequencers. They are also extended or redesigned when further development of the instrument is instigated, for example, development of the precision modes of nulling and differential phase from basic control achieved fro visibility science. Sequencing almost always was implemented first in scripting languages, allowing rapid prototyping and development[6].

2.2. Hardware

The above approach to software development: capitalizing on WMKO experience and making systems modular for ease of development and maintenance was also repeated for hardware. As far as possible we developed the control systems using hardware already in use at WMKO, or already in their planned upgrade path. We also repeated use of hardware between EPICS and RTC systems. This was fortunately aided by the fact that similar hardware and development environments to those previously used by WMKO were already in use at JPL for test bed development of the RTC tool kit. Thus, all our subsystems are deployed as Power PC control in a VME environment using the VxWorks operating system. We make extensive use of Industry Pack modules for D to A and A to D, and for digital IO, for example, and have reused and developed WMKO systems based on PMAC intelligent controllers. Many of the physical actuators used in the interferometer are identical to those already in use at WMKO, Newport 850G positioners, for example.

3. SUB-SYSTEM CONTROL

3.1. RTC controlled "fast" sub-systems

All the hard real time sub-systems in the interferometer control system that must run at "fast" rates (i.e. more than about 100Hz) are built using the JPL developed RTC control system[5]. This control system development tool has been developed at JPL as part of the interferometer development and test bed program, of which the Keck interferometer project is part. Thus, the framework has been developed with interferometer control especially in mind. The framework has been developed using C++, and Java, in both the VxWorks real time OS, and UNIX and Linux OS regimes. The framework is inherently object oriented.

The framework contains the following parts:

- Real time control system frameworks in C++ for VxWorks. The frameworks include real time state machines, device drivers, and predefined interfaces for commanding and telemetry.
- Servers for distributing telemetry in C++ under UNIX/Linux and an archiving system for capturing telemetry to disk, also in C++ for UNIX/Linux.
- An extensive relational data base system for configuration management, including a java based GUI, used to maintain configurable parameters for the real time systems making for easy deployment of multiple copies of sub-system controllers with differently configured properties. Reconfiguration can occur at run time, and even "on the fly".
- A flexible engineering GUI development tool kit developed in Java for UNIX/Linux/Windows, allowing on the fly configurable textual and graphical display of telemetry, and easy implementation of graphical command interfaces.
- Timing and scheduling control mechanisms and hardware for the real time processes.

In the following sub-sections we give some details of the real time sub-systems developed for the Keck Interferometer using this framework. Each description is for a single instance of each sub-system, of which there are generally several

in the interferometer system. All sub-systems are controlled independently by a single instance of their controlling software system, loaded with parameters from the configuration data base.

3.2. Delay lines

The delay lines are used to provide variable optical path length in each beam train from telescope to detectors, in response to sidereal motion of the target objects, and to provide an actuator for the fringe tracker system. They are implemented as a 4 stage nested servo system (similar to the delay lines used at the Palomar Testbed Interferometer[7]): an inner loop controlling a fast PZT mounted small mirror, two intermediate loops controlling voice coils actuating entire optics assemblies, and an outer loop controlling a micro-stepper motor that drives the whole assembly along rails. The inner loop responds to feed back error signals measuring the actual position of the assembly along its rail path using laser metrology, with a bandwidth of about 500Hz. Outer loops respond to this error at progressively lower bandwidths. Various targets are provided for the position of the assembly with regard to the laser metrology: a sidereal target from a higher level sequencer; a "continuous term" metrology signal, measuring the total path length of the beam train of which this delay line forms a part, from the fringe tracker to the telescope; and a measurement of the change in optical path length caused by motion of the telescope optics, measured using accelerometers mounted on the telescope. The closed loop target from the fringe tracker is either an actual error signal from a measurement of fringe position, or a search target if the tracker detects no fringes.

In common with all the real time control sub-systems, the RTC framework has been used to implement a state machine that allows the delay lines to be either "idle" (measuring sensor input, but not moving closed loop), or tracking. In tracking, the sub-system can be in a variety of states, in this case: slewing (moving rapidly to a new target), or locked (closed loop tracking on error signal). The RTC framework allows easy implementation of the state machine that defines these states and controls transition between them based on user input and sensor values.

3.3. Fringe trackers

The fringe tracker is implemented as two separate processes running on the same CPU. One process controls the camera readout, and one performs the servo calculations. The camera readout process is very simple, loading the appropriate clock pattern to read out the array and setting up the rate at which the pattern is applied, then placing the results of the readout into local memory. The servo process obtains the read out pixels from the camera, and sends a calculated target to the delay line sub-systems to keep the fringe packet centered on the beam combiner. The calculation involves determining a fringe phase from a white light pixel from one output of the beam combiner, and a fringe position group delay determined from a dispersed spectrum from the other output, then combining these two. Details of this process can be found in Vashist[8]. The state machine for the fringe tracker allows for idle and tracking states. In tracking the process can be searching (sending a linear spiral search target to the delay lines instead of an error signal), semi-lock (confirming the existence of suspected fringes), or lock (confirmed detection of fringes, and servoing delay lines).

3.4. Angle trackers

The angle trackers also have a camera readout process and a servo process. The camera readout process is virtually identical to that of the fringe tracker. The servo process determines a position for the target image on the camera and calculates an error signal to be sent to the tip-tilt metrology sub-system to keep the target centered on the detector. The target is determined using either a centroid algorithm, in initial wide angle mode, or a quad cell algorithm once the target has been acquired. The state machine allows for idle and tracking states, and for search, semi-lock and lock in tracking. In addition to providing an error signal to the tip-tilt system, the angle tracker also offloads the absolute tip-tilt mirror position to the telescope tracking systems (in the case of the Keck telescopes, this means the AO system) at a lower rate than the main servo to keep the mirrors close to center.

3.5. Tip-tilt metrology

The tip-tilt metrology sub-system performs a higher bandwidth control of the internal beam train tip-tilt alignment than can be achieved by the angle tracker. The angle tracker works using detected starlight from the telescope, so is limited by the photon rate of the object. The tip-tilt sub-system uses a laser beacon near the beam combiner at the fringe tracker to illuminate a detector near the telescope at the other end of the beam train. Tip and tilt are calculated from the laser

spot position on the detector, and fed back to a control mirror to keep the spot immobile. Targets are also added in from the angle trackers as mentioned above to keep the star centered on its detector (and hence ensure that light is correctly falling on the fringe trackers). The state machine for this sub-system uses the idle and track states. In the track state the state machine is very simple, allowing for a lock state if there is laser light falling on the detector.

4. SEQUENCING

The sub-systems described above run in hard real time, controlled by hardware clocks and with scheduling that requires the tasks complete in less time than required by the servo rate. This ensures optimum performance of the closed loop servos that are the primary concern of these sub-systems. These systems interface to a higher level of control that does not have to be run in hard real time, and thus has not been implemented in the RTC frame work. We refer to this level of control as sequencing, and it satisfies several requirements for the interferometer control system as indicated above: making the various subsystems act in concert, controlling internal calibrations for a single target object, and sequencing through a set of science and calibrator target objects.

As indicated above the sequencing systems are implemented as a hierarchical structure. Fig. 1 shows the hierarchy of the sequencing components. The subsystem sequencers are instantiated as one per real time sub-system, and control their individual subsystem via commands and monitored telemetry. They ensure that the more generic commands received from the interferometer sequencer are correctly interpreted in the command set of the real time sub-systems, and that those systems enter the expected state as a consequence of the commands. They allow for faulted states if the sub-systems do not perform in the expected manner. The interferometer sequencer contains several modules that perform the correct coordination of the real time subsystems (and the telescope systems) to achieve a science observation, including sequencing through internal calibrations. These internal calibrations include commands to "slow" subsystems implemented in EPICS (e.g. commands to close shutters) as well as "fast" systems implemented in RTC.

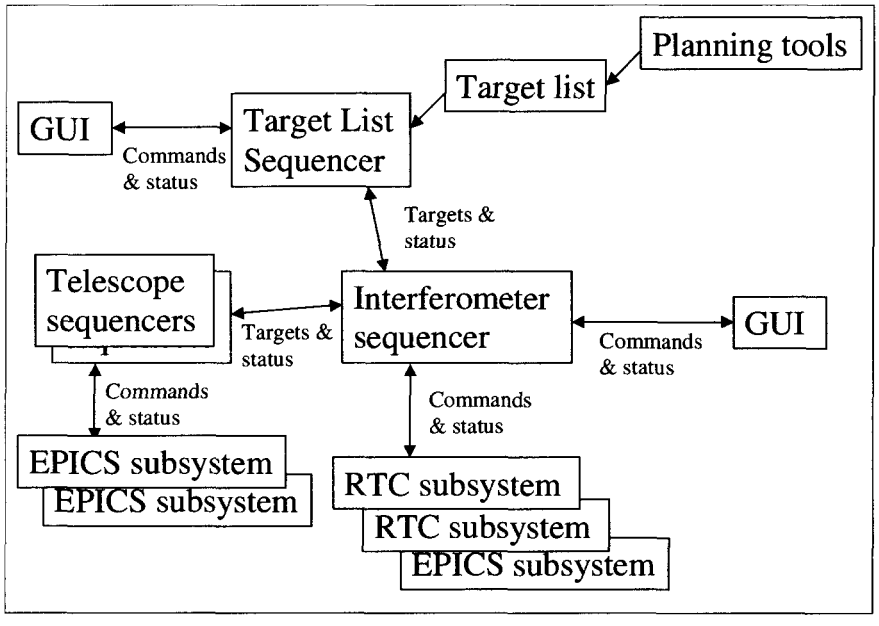

Fig. 1 Sequencing hierarchy

At the highest level the target list sequencer controls the interferometer sequencer and performs the function of cycling between science and calibration targets based on a target list supplied by planning tools[9].

The interferometer sequencer, sub-system sequencers and target list sequencer are being implemented using the "Rhapsody" tool from I-Logix[10]. This is a UML based tool with VPE interface which generates C++ code in an object oriented frame work. As mentioned, prototyping for sequencers is performed using scripting languages, and at present the target sequencer is implemented in Tcl/Tk. Telescope sequencers and telescope subsystem sequencers are being

implemented in the EPICS state notation language frame work. All these sequencers are currently implemented under the UNIX operating system.

5. AUTO-ALIGNMENT

As indicated above, an optical system as complicated as the Keck interferometer must have an auto-alignment system to allow quick reconfiguration, alignment checks and improvement during observations, improved accuracy and repeatability of alignment and improved human interface to optical alignment where the optics are located in a hostile environment (14000ft altitude). Details of the auto-alignment system are given by Van Belle, et al.[11].

The auto-alignment control system is implemented in a very similar manner to the main instrument control system. Individual sub-systems (e.g. 2 axis mirror controllers) are implemented in EPICS, often with self contained sequencers for automated start up, slew to position, etc. These systems are real time and run under the VxWorks operating system. At a higher level, there are sub-system sequencers, in this case implemented in the EPICS State Notation Language (SNL) frame work, controlling the external command and telemetry interface. Above these is a further layer of sequencing that controls related subsystems to achieve auto-alignment on a single optic, again in EPICS SNL. A typical sequence involves turning on a stimulus, detecting a centroid for a target on a camera, and servoing the optic position to center the target. At the highest level, a sequencer is provided to step through all the optical elements in a beam train (or all those that need to be touched after, say, a reconfiguration) and perform an alignment. At present this top level sequencing is implemented as a prototype in Tcl/Tk, but will shortly be ported to Rhapsody, with a java GUI.

6. INTERFACE CONTROL

It can be seen from the foregoing that the total number of interfaces in the interferometer control system is large, and that they are potentially of a diverse nature. There is inter-communication between systems running under different operating systems – VxWorks and UNIX/Linux, and systems built using two very dissimilar frame works – RTC and EPICS, and there are a variety of languages used – C++, C, java, scripting languages. However, the interfaces can be placed into three groups, as described in the following three paragraphs.

As well as having a native interface called channel access, the EPICS systems in use at WMKO have a legacy communications layer (KTL) based on keywords for both command and telemetry. Tcl/Tk has been extended to interface to KTL keywords. This layer is well engineered and flexible, allowing easy communication across operating systems and between CPUs, so we wished to continue to use it in the interferometer EPICS systems. Thus, all the interfaces between EPICS systems in the interferometer, and between interferometer EPICS systems and telescope EPICS systems are implemented in the KTL frame work.

The RTC sub-systems at the real time (VxWorks) level (e.g. delay lines, fringe tracker) have a built in inter-processor communications protocol (IPC) that is used, for example, for communication of fringe tracker targets to the delay lines. IPC works between CPUs on the same VME back plane, and between CPUs in separate VME crates provided the physical distance is not large (this is important in the interferometer, where the CPUs for the interferometer are far from those controlling telescope systems).

For communication between the non-real time parts of RTC (e.g. GUIs, configuration data base), and from these parts to the real time sub-systems, a communications layer based on CORBA is provided. CORBA is inherently object oriented, and so fits well in the RTC design philosophy. CORBA and its application to RTC is beyond the scope of this article (see Lockhart[5] for more information), but CORBA is designed to make communication interfaces more transparent by confining user accessible parts of the interface to a C++ like definition called IDL. CORBA ORBs are available for all the languages (C++, Java, Tcl, Python) and all the operating systems (VxWorks, Unix/Linux) in use in the interferometer control system. We were thus able to construct a common interface based on IDL definitions for all RTC based systems. For example, one definition of telemetry exists that is used by all RTC based systems to report collected science data, systems health and system status. Use of CORBA thus makes interfacing between heterogeneous systems more straightforward and helps solve the interface complexity issues for the RTC based systems.

Thus, controlling interface complexity in the interferometer is largely reduced to providing a simple interface between CORBA and keyword based systems. Fortunately, much of this interface can also be confined to the sequencer level,

and specifically most communication can be conveniently routed through the interferometer sequencer without loss of performance or introduction of data flow complexity. We have thus provided both a CORBA based interface layer and a keyword based interface layer as parts of the Rhapsody frame work used to construct the interferometer sequencer. Thus, for example, this sequencer can command RTC systems based on CORBA method calls, and at the same time send keyword writes containing target information to telescope sequencers. No direct translation of keyword to CORBA systems is provided as the information needs in all cases to be processed through the sequencer, for it to provide its sequencing functionality.

There are a very few cases where it does not make sense to route commands and data passing between RTC and EPICS based systems through the sequencer layer. An example is the mirror position offload commands generated by the angle tracker RTC sub-system being sent to the EPICS based AO sub-system. In all these cases the interface is very simple, requiring only that a simple position, or position pair be passed (in the form of a primitive integer or float type) from CORBA to KTL or vice versa. In these cases we have provided a simple thin translation layer to implement the communication. These layers are implemented on UNIX to avoid complications of scheduling in VxWorks, and are generic enough to handle the simple interface requirements of all these special cases.

7. STATUS

At the present time the control system for the Keck Interferometer has been developed to the extent needed to perform traditional visibility science measurements using the two Keck telescopes. The following sub-systems have been implemented: two delay lines (one for each beam train), two angle trackers, two tip-tilt metrology systems, and one fringe tracker. In addition, sequencing of these sub-systems has been implemented and this sequencer interfaces to the telescope systems through a telescope sequencer to provide target information. Constant term metrology is used to measure the total path length variations along the whole beam train, and accelerometers are used to do likewise for the telescope optics. All optical elements that are routinely moved during set up, alignment, or calibrations (e.g. mirrors, shutters, retro-reflectors, etc) have been automated and are controlled by either the interferometer or auto-alignment sequencers, such that the interferometer can be run entirely remotely during normal operation. Extensive user interfaces have been provided in java and Tcl/Tk that also allow easy remote operation. Indeed the full interferometer has often been operated in internal calibration mode from JPL during day time testing.

Near term developments include: automated control of the Long Delay Lines to provide larger amounts of (fixed) optical path differencing to allow extended sky coverage, and inclusion of these into the auto-alignment system; development of the nulling beam combiner; and development of the differential phase mode of the instrument.

ACKNOWLEDGEMENTS

The work reported here was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration, and at the W. M. Keck Observatory, California Association for Research in Astronomy. Funding for the Keck Interferometer is provided by the National Aeronautics and Space Administration.

REFERENCES

1. See http://huey.jpl.nasa.gov/keck/, http://planetquest.jpl.nasa.gov/, http://origins.jpl.nasa.gov/
2. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer: progress report", in Interferometry in Optical Astronomy, Proc. SPIE, 4006, 2000.
3. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer update", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
4. See http://mesa53.lanl.gov/lansce8/EPICS
5. T. Lockhart, "RTC: a distributed real-time control system toolkit", Proc. SPIE, 4848, 2002 (this conference).
6. L. Reder, et al., "Using scripting languages in optical interferometry", Proc. SPIE, 4848, 2002 (this conference).
7. M. M. Colavita, et al., "The Palomar testbed interferometer", Ap.J., **510**, pp. 505-521, 1999

8. G. Vasisht et al., "Performance and verification of the Keck interferometer fringe detection and tracking system: FATCAT", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
9. See http://isc.caltech.edu/KISupport.
10. See http://www.ilogix.com.
11. G.T. Van Belle, et al., "Keck interferometer autoaligner", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).

Overview of the control system for the Keck Interferometer

A. J. Booth*[a], G. Eychaner*[a], E. Hovland*[a], R. L. Johnson Jr.*[a], W. Lupton**[b], A. Niessner*[a], D. Palmer*[a], L. J. Reder*[a], A. C. Rudeen**[b], R. F Smythe*[a], K. Tsubota**[b]
[a]Jet Propulsion Laboratory, California Institute of Technology, [b]W. M. Keck Observatory, California Association for Research in Astronomy

ABSTRACT

The Keck Interferometer links the two 10m Keck Telescopes located atop Mauna Kea in Hawaii. It is the first 10m class, fully AO equipped interferometer to enter operation. Further, it is the first large interferometer designed to be handed over from a design and implementation team to a separate operations team, and be used by astronomers who are not interferometer specialists. As such it offers unique challenges in reducing an extremely complex and powerful system to an apparently simple user interface, and providing a well engineered system that can be maintained by people who did not develop it.

This paper gives an overview of the control system that has been implemented for single baseline operation of the instrument, and indicates how this will be extended to allow control of the future modes of the instrument (nulling, differential phase and astrometry).

The control system has several parts. One is for control of "slow" sub-systems which is based in the EPICS architecture, already ubiquitous at the Keck Observatory. Another, used to control real time sub-systems, is based on a new infrastructure developed at JPL, programmed in C++, Java, and using CORBA for communication. This infrastructure has been developed specifically with the problems of interferometry control in mind and is used in JPL's flight testbeds as well as the Keck Interferometer. Finally, a user interface and high level control layer is in development using a variety of tools including UML based modeling in the Rhapsody tool (using CORBA), Java, and Tcl/Tk for prototyping.

Keywords: Interferometry, control system, real time control

1. INTRODUCTION

1.1. Overview

The Keck Interferometric Array, as proposed, would use Michelson combiners to link the two Keck 10m telescopes along with four proposed 1.8m outrigger telescopes. The two Kecks provide baselines of 85 m in the NE direction. With the addition of the outriggers, a range of baselines from 30 m to 135 m would be provided. For highest efficiency, all of the telescopes are phased: the Kecks use adaptive optics, while the outriggers would use fast tip/tilt corrections; for off-source cophasing, a Cophasing among baselines is provided using active fringe tracking and a cophasing reference to the dual-star module will be installed at each telescope to bring the light from all four telescopes to the beam combining lab. The back-end science instruments of the interferometer include three-way combiners at 1.5-2.4 μm for astrometry, traditional visibility science, and cophasing; a proposed multi-way combiner at 1.5-5 μm; and a nulling combiner at 10 μm in development. The Keck Interferometer is a first class, fully AO equipped interferometer to enter operation and is described in detail by Colavita et al.[2] The Keck Interferometer is an element in NASA's Origins Program[1].

...nasa.gov; phone 1 818 354 1200; fax 1 818 393 4...aboratory, MS171-113, ...CA 91109; **ktsubota@keck.hawai...5 7887; fax 1 808 885 4464; ...hway, Kamuela, H

Overview of the control system for the Keck Interferometer

A. J. Booth[*a], G. Eychaner*[a], E. Hovland*[a], R. L. Johnson Jr.*[a], W. Lupton**[b], A. Niessner*[a], D. Palmer*[a], L. J. Reder*[a], A. C. Rudeen**[b], R. F Smythe*[a], K. Tsubota**[b]
[a]Jet Propulsion Laboratory, California Institute of Technology, [b]W. M. Keck Observatory, California Association for Research in Astronomy

ABSTRACT

The Keck Interferometer links the two 10m Keck Telescopes located atop Mauna Kea in Hawaii. It is the first 10m class, fully AO equipped interferometer to enter operation. Further, it is the first large interferometer designed to be handed over from a design and implementation team to a separate operations team, and be used by astronomers who are not interferometer specialists. As such it offers unique challenges in reducing an extremely complex and powerful system to an apparently simple user interface, and providing a well engineered system that can be maintained by people who did not develop it.

This paper gives an overview of the control system that has been implemented for the single baseline operation of the instrument, and indicates how this will be extended to allow control of the future modes of the instrument (nulling, differential phase and astrometry).

The control system has several parts. One is for control of "slow" sub-systems, which is based in the EPICS architecture, already ubiquitous at the Keck Observatory. Another, used to control hard real time sub-systems, is based on a new infrastructure developed at JPL, programmed in C++, Java, and using CORBA for communication. This infrastructure has been developed specifically with the problems of interferometric control in mind and is used in JPL's flight testbeds as well as the Keck Interferometer. Finally, a user interface and high level control layer is in development using a variety of tools including UML based modeling in the Rhapsody tool (using C++ and CORBA), Java, and Tcl/Tk for prototyping.

Keywords: Interferometry, control system, real time control

1. INTRODUCTION

1.1. Overview

The Keck Interferometric Array, as proposed, would use Michelson combination among the two Keck 10m telescopes along with four proposed 1.8m outrigger telescopes. The two Kecks provide a baseline of 85 m in the NE direction. With the addition of the outriggers, a range of baselines from 30 m to 135 m could be provided. For highest efficiency, all of the telescopes are phased: the Kecks use adaptive optics, while the outriggers would use fast tip/tilt correction. Cophasing among baselines is provided using active fringe tracking and active delay lines; for off-source cophasing, a dual-star module will be installed at each telescope to bring the light from the source and a cophasing reference to the beam combining lab. The back-end science instruments of the interferometer include two-way combiners at 1.5-2.4 μm for astrometry, traditional visibility science, and cophasing; a proposed multi-way imaging combiner at 1.5-5 μm; and a nulling combiner at 10 μm in development. The Keck Interferometer is the first 10m class, fully AO equipped interferometer to enter operation and is described in detail by Colavita and Wizinowich[2,3]. The Keck Interferometer is an element in NASA's Origins Program[1].

* Andrew.J.Both@jpl.nasa.gov; phone 1 818 354 1200; fax 1 818 393 4357; Jet Propulsion Laboratory, MS171-113, 4800 Oak Grove Drive, Pasadena, CA 91109; **ktsubota@keck.hawaii.edu; phone 1 808 885 7887; fax 1 808 885 4464; W. M. Keck Observatory, 65-1120 Mamalahoa Highway, Kamuela, HI 96743

The science goals of the interferometer are the use of nulling interferometry to characterize zodiacal dust emission from nearby solar type stars as a precurser to the SIM and TPF planet finder missions (nulling mode); to detect hot Jupiters (Jovian sized planets in close orbit around their parent stars) using differential phase signals at varying detector wavelengths (DP mode); astrometric searches for planets down to Uranus mass around nearby stars (astrometry mode); and general purpose high resolution astronomical imaging of a variety of sources using the full array (imaging mode). For a detailed description of the science goals of the instrument see Colavita and Wizinowich[2].

1.2. Control system

The control system for the Keck Interferometer is large and complicated. There are many parts and many interfaces, and it is being developed by two physically distant groups – JPL and WMKO, so there is ample potential for development of a system that is difficult to use and maintain. In addition, the system must be handed over from the development group (which for the interferometer itself, as opposed to telescope systems, is dominated by JPL personnel) to the operations group, which will contain none of the JPL developers. Further, the instrument must be of facility class, such that it may be used by astronomers who are not interferometry specialists. Thus, it is requirement that the instrument be, in fact, easy to use and maintain. In addition there are very tight requirements on the performance of the system to reach the challenging science goals of, for example, nulling detection of exo-zodiacal emission and differential phase measurements of hot Jupiters. These requirements are not easy to achieve.

The main components of the control system are:

- **Angle trackers,** which keep the light from the telescopes falling on the entrance aperture to the fringe trackers, and keep the wavefronts that are being combined to form interference fringes parallel.
- **Delay lines,** which keep the optical path difference between any two telescope beams that are being combined the same to within very tight tolerances whilst tracking the moving delay cause by the sidereal motion of the target object.
- **Metrology,** which monitors and corrects for path length and tilt variations in the optical beam trains of the interferometer from the telescopes to the detectors
- **Fringe trackers,** which not only measure the fringe position and provide feed back to delay lines to maintain lock on the interference, but also provide the measurements for phase and visibility that form the science observations.
- **Nulling and imaging cameras,** which will measure the fringe quantities of interest to nulling and imaging science.
- **Telescope systems,** being the telescope and dome control themselves, and also target acquisition, adaptive optics (AO) control, and coudé beam train control (particularly for off axis beam trains used for the secondary star feed).

For most of these sub-systems, several copies of functionally identical, but differently configured versions of the sub-systems are used together. For example, in a full imaging mode deployment there would be twelve delay lines operating simultaneously. There are numerous interfaces between the different sub-systems leading to a complicated data flow in the instrument. For example, as mentioned above, the fringe trackers measure the position of the interference fringes formed by combining the light from two telescopes. This position is fed back to the delay lines so as to keep the center of the interference fringe packet on the beam combiner, to maintain the highest fringe visibility. As another example the angle trackers feed back to the AO systems to keep the target object aligned with the detectors.

The tolerances on the performance of the sub-systems are very demanding. Servos must run at fast rates, for example the delay lines internally servo their positions against a metrology source at 5kHz. The angle trackers and fringe trackers need to read out their camera hardware at close to the maximum allowable rate to provide enough bandwidth for fringe tracking control and angle tracking control to overcome the variability induced by the earth's atmosphere and the internal variations in the instrument (fringe tracker frames at up to 500Hz and angle tracker frames at up to 100Hz). The servo control must also be very precise. The fringe tracking must control the fringe position to much better than a wavelength of light (so to better than 100nm or so), even though the delay lines are moving over 10s of meters during an observation.

To successfully make science measurements with the instrument there must be a high degree of automation for the control system. This also naturally leads to a control system that is easy to use since the user need not have a detailed

knowledge of the internal workings of the subsystems to achieve a set of science observations. The main reason for this automation requirement is that interferometry science requires high levels of careful calibration, thus we require:

- Fast switching between taking science data and taking internal calibration data.
- Fast switching between science target objects and objects of known visibility and phase for external fringe calibration.
- Consistency of calibration processes from targets to calibrator objects and in set up of target and calibrators.

These requirements demand automation at several levels:

- Automated sequencing of sub-system operation within a subsystem.
- Automated sequencing of sub-systems for coordinated operation of subsystems.
- Automated sequencing of targets presented to the instrument.
- Automated optical alignments of system and sub-system optical elements

This last automation requirement arises because the interferometer has many optical elements in each beam train from telescope to detector (many 10s of elements), and the total optical path is very long (many 10s of meters). Thus, manual alignment would be too time consuming and also not of sufficient accuracy.

2. DESIGN APPROACH

2.1. Software

The design approach taken for the interferometer control system was driven by the above complexities and requirements. In this section we give an overview of the thinking behind the current and planned implementations.

At WMKO extensive use is already made of the EPICS[4] control framework. Many observatory sub-systems, including telescope control, are written using EPICS, and there are additional tool sets developed by WMKO which are layered on top of EPICS, for example a keyword interface layer (KTL). There thus exists a large body of expertise and knowledge at WMKO in the use of EPICS and many control systems are built on thoroughly tested device control modules built in EPICS. It was therefore decided wherever possible, to make use of existing EPICS developments at WMKO for control of interferometer sub-systems, and to use WMKO expertise to extend existing control systems where necessary to control interferometer sub-systems that are similar to these existing control systems. This not only builds on existing infrastructure, but also helps the handover process as WMKO personnel are already familiar with the details of control systems for these interferometer elements, even if they were developed by JPL personnel. Care was taken to also make the development and deployment environment of EPICS systems at JPL adhere to WMKO policy and procedures.

Much of the interferometer's control system must run at rates too high to be implemented using EPICS. For these parts of the system we chose to implement control using the JPL developed RTC tool set[5]. The use of this tool set is described more fully below, however, here we note that the tool set allows a modular approach to design of the control system. We have many subsystems that are very similar in concept (for example, angle trackers and fringe trackers both read out a camera at high rate, compute a target, and send the target to another subsystem for actuation); we have many copies of the same type of subsystem, as noted above for example for the 12 delay lines needed in imaging mode. The object oriented nature of the RTC tool set allows easy modularisation of the elements of the control system to achieve this repeatability in design and in instantiation.

In addition to being implemented as a modular system, the control system is also very hierarchical. This particularly applies to the sequencing type of control for the system and for the sub-systems. Sequencing occurs at all levels within in the control system: At the lowest level, the sub-systems sequence their internal states based on low level input from sensors. For example, is there light in the detector, then start to track the fringes. At an intermediate level, sub-systems must be coordinated to act in concert. For example the observational target must be acquired by the angle trackers before the fringe trackers can try to track fringes. Also, the control system must cycle through various states of the sub-systems to achieve an observing sequence with the required internal calibrations. Finally at the highest level the control system must cycle through a predefined list of targets and calibration sources to achieve scientifically interesting measurements. The hierarchical nature of this sequencing allows us to implement it in a modular and hierarchical way, with separate control objects for each level of the hierarchy, linked by common format interfaces. This allows for

separate development and testing, and implementation of prototypes for various levels along side more fully developed systems at other levels.

For sub-systems developed within the EPICS framework, we also employed hierarchical development, layering sequencers on top of collections of individual sub-system modules, to implement, for example, the auto-alignment system.

We made extensive use of prototyping and iterative development of sub-systems and sequencers. Sub-systems are developed as stand alone modules, then interfaced to other subsystems and sequencers. They are also extended or redesigned when further development of the instrument is instigated, for example, development of the precision modes of nulling and differential phase from basic control achieved fro visibility science. Sequencing almost always was implemented first in scripting languages, allowing rapid prototyping and development[6].

2.2. Hardware

The above approach to software development: capitalizing on WMKO experience and making systems modular for ease of development and maintenance was also repeated for hardware. As far as possible we developed the control systems using hardware already in use at WMKO, or already in their planned upgrade path. We also repeated use of hardware between EPICS and RTC systems. This was fortunately aided by the fact that similar hardware and development environments to those previously used by WMKO were already in use at JPL for test bed development of the RTC tool kit. Thus, all our subsystems are deployed as Power PC control in a VME environment using the VxWorks operating system. We make extensive use of Industry Pack modules for D to A and A to D, and for digital IO, for example, and have reused and developed WMKO systems based on PMAC intelligent controllers. Many of the physical actuators used in the interferometer are identical to those already in use at WMKO, Newport 850G positioners, for example.

3. SUB-SYSTEM CONTROL

3.1. RTC controlled "fast" sub-systems

All the hard real time sub-systems in the interferometer control system that must run at "fast" rates (i.e. more than about 100Hz) are built using the JPL developed RTC control system[5]. This control system development tool has been developed at JPL as part of the interferometer development and test bed program, of which the Keck interferometer project is part. Thus, the framework has been developed with interferometer control especially in mind. The framework has been developed using C++, and Java, in both the VxWorks real time OS, and UNIX and Linux OS regimes. The framework is inherently object oriented.

The framework contains the following parts:

- Real time control system frameworks in C++ for VxWorks. The frameworks include real time state machines, device drivers, and predefined interfaces for commanding and telemetry.
- Servers for distributing telemetry in C++ under UNIX/Linux and an archiving system for capturing telemetry to disk, also in C++ for UNIX/Linux.
- An extensive relational data base system for configuration management, including a java based GUI, used to maintain configurable parameters for the real time systems making for easy deployment of multiple copies of sub-system controllers with differently configured properties. Reconfiguration can occur at run time, and even "on the fly".
- A flexible engineering GUI development tool kit developed in Java for UNIX/Linux/Windows, allowing on the fly configurable textual and graphical display of telemetry, and easy implementation of graphical command interfaces.
- Timing and scheduling control mechanisms and hardware for the real time processes.

In the following sub-sections we give some details of the real time sub-systems developed for the Keck Interferometer using this framework. Each description is for a single instance of each sub-system, of which there are generally several

in the interferometer system. All sub-systems are controlled independently by a single instance of their controlling software system, loaded with parameters from the configuration data base.

3.2. Delay lines

The delay lines are used to provide variable optical path length in each beam train from telescope to detectors, in response to sidereal motion of the target objects, and to provide an actuator for the fringe tracker system. They are implemented as a 4 stage nested servo system (similar to the delay lines used at the Palomar Testbed Interferometer[7]): an inner loop controlling a fast PZT mounted small mirror, two intermediate loops controlling voice coils actuating entire optics assemblies, and an outer loop controlling a micro-stepper motor that drives the whole assembly along rails. The inner loop responds to feed back error signals measuring the actual position of the assembly along its rail path using laser metrology, with a bandwidth of about 500Hz. Outer loops respond to this error at progressively lower bandwidths. Various targets are provided for the position of the assembly with regard to the laser metrology: a sidereal target from a higher level sequencer; a "continuous term" metrology signal, measuring the total path length of the beam train of which this delay line forms a part, from the fringe tracker to the telescope; and a measurement of the change in optical path length caused by motion of the telescope optics, measured using accelerometers mounted on the telescope. The closed loop target from the fringe tracker is either an actual error signal from a measurement of fringe position, or a search target if the tracker detects no fringes.

In common with all the real time control sub-systems, the RTC framework has been used to implement a state machine that allows the delay lines to be either "idle" (measuring sensor input, but not moving closed loop), or tracking. In tracking, the sub-system can be in a variety of states, in this case: slewing (moving rapidly to a new target), or locked (closed loop tracking on error signal). The RTC framework allows easy implementation of the state machine that defines these states and controls transition between them based on user input and sensor values.

3.3. Fringe trackers

The fringe tracker is implemented as two separate processes running on the same CPU. One process controls the camera readout, and one performs the servo calculations. The camera readout process is very simple, loading the appropriate clock pattern to read out the array and setting up the rate at which the pattern is applied, then placing the results of the readout into local memory. The servo process obtains the read out pixels from the camera, and sends a calculated target to the delay line sub-systems to keep the fringe packet centered on the beam combiner. The calculation involves determining a fringe phase from a white light pixel from one output of the beam combiner, and a fringe position group delay determined from a dispersed spectrum from the other output, then combining these two. Details of this process can be found in Vashist[8]. The state machine for the fringe tracker allows for idle and tracking states. In tracking the process can be searching (sending a linear spiral search target to the delay lines instead of an error signal), semi-lock (confirming the existence of suspected fringes), or lock (confirmed detection of fringes, and servoing delay lines).

3.4. Angle trackers

The angle trackers also have a camera readout process and a servo process. The camera readout process is virtually identical to that of the fringe tracker. The servo process determines a position for the target image on the camera and calculates an error signal to be sent to the tip-tilt metrology sub-system to keep the target centered on the detector. The target is determined using either a centroid algorithm, in initial wide angle mode, or a quad cell algorithm once the target has been acquired. The state machine allows for idle and tracking states, and for search, semi-lock and lock in tracking. In addition to providing an error signal to the tip-tilt system, the angle tracker also offloads the absolute tip-tilt mirror position to the telescope tracking systems (in the case of the Keck telescopes, this means the AO system) at a lower rate than the main servo to keep the mirrors close to center.

3.5. Tip-tilt metrology

The tip-tilt metrology sub-system performs a higher bandwidth control of the internal beam train tip-tilt alignment than can be achieved by the angle tracker. The angle tracker works using detected starlight from the telescope, so is limited by the photon rate of the object. The tip-tilt sub-system uses a laser beacon near the beam combiner at the fringe tracker to illuminate a detector near the telescope at the other end of the beam train. Tip and tilt are calculated from the laser

spot position on the detector, and fed back to a control mirror to keep the spot immobile. Targets are also added in from the angle trackers as mentioned above to keep the star centered on its detector (and hence ensure that light is correctly falling on the fringe trackers). The state machine for this sub-system uses the idle and track states. In the track state the state machine is very simple, allowing for a lock state if there is laser light falling on the detector.

4. SEQUENCING

The sub-systems described above run in hard real time, controlled by hardware clocks and with scheduling that requires the tasks complete in less time than required by the servo rate. This ensures optimum performance of the closed loop servos that are the primary concern of these sub-systems. These systems interface to a higher level of control that does not have to be run in hard real time, and thus has not been implemented in the RTC frame work. We refer to this level of control as sequencing, and it satisfies several requirements for the interferometer control system as indicated above: making the various subsystems act in concert, controlling internal calibrations for a single target object, and sequencing through a set of science and calibrator target objects.

As indicated above the sequencing systems are implemented as a hierarchical structure. Fig. 1 shows the hierarchy of the sequencing components. The subsystem sequencers are instantiated as one per real time sub-system, and control their individual subsystem via commands and monitored telemetry. They ensure that the more generic commands received from the interferometer sequencer are correctly interpreted in the command set of the real time sub-systems, and that those systems enter the expected state as a consequence of the commands. They allow for faulted states if the sub-systems do not perform in the expected manner. The interferometer sequencer contains several modules that perform the correct coordination of the real time subsystems (and the telescope systems) to achieve a science observation, including sequencing through internal calibrations. These internal calibrations include commands to "slow" subsystems implemented in EPICS (e.g. commands to close shutters) as well as "fast" systems implemented in RTC.

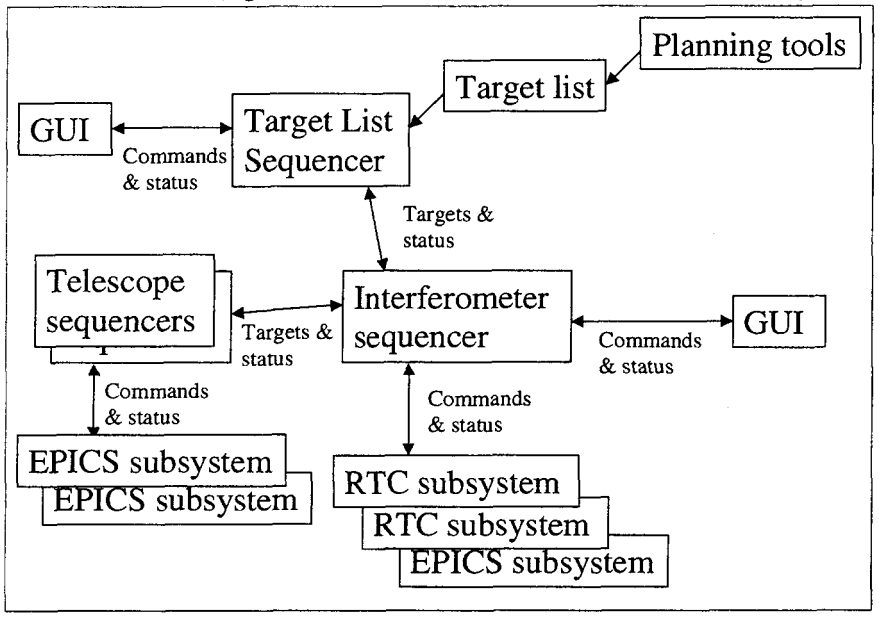

Fig. 1 Sequencing hierarchy

At the highest level the target list sequencer controls the interferometer sequencer and performs the function of cycling between science and calibration targets based on a target list supplied by planning tools[9].

The interferometer sequencer, sub-system sequencers and target list sequencer are being implemented using the "Rhapsody" tool from I-Logix[10]. This is a UML based tool with VPE interface which generates C++ code in an object oriented frame work. As mentioned, prototyping for sequencers is performed using scripting languages, and at present the target sequencer is implemented in Tcl/Tk. Telescope sequencers and telescope subsystem sequencers are being

implemented in the EPICS state notation language frame work. All these sequencers are currently implemented under the UNIX operating system.

5. AUTO-ALIGNMENT

As indicated above, an optical system as complicated as the Keck interferometer must have an auto-alignment system to allow quick reconfiguration, alignment checks and improvement during observations, improved accuracy and repeatability of alignment and improved human interface to optical alignment where the optics are located in a hostile environment (14000ft altitude). Details of the auto-alignment system are given by Van Belle, et al.[11].

The auto-alignment control system is implemented in a very similar manner to the main instrument control system. Individual sub-systems (e.g. 2 axis mirror controllers) are implemented in EPICS, often with self contained sequencers for automated start up, slew to position, etc. These systems are real time and run under the VxWorks operating system. At a higher level, there are sub-system sequencers, in this case implemented in the EPICS State Notation Language (SNL) frame work, controlling the external command and telemetry interface. Above these is a further layer of sequencing that controls related subsystems to achieve auto-alignment on a single optic, again in EPICS SNL. A typical sequence involves turning on a stimulus, detecting a centroid for a target on a camera, and servoing the optic position to center the target. At the highest level, a sequencer is provided to step through all the optical elements in a beam train (or all those that need to be touched after, say, a reconfiguration) and perform an alignment. At present this top level sequencing is implemented as a prototype in Tcl/Tk, but will shortly be ported to Rhapsody, with a java GUI.

6. INTERFACE CONTROL

It can be seen from the foregoing that the total number of interfaces in the interferometer control system is large, and that they are potentially of a diverse nature. There is inter-communication between systems running under different operating systems – VxWorks and UNIX/Linux, and systems built using two very dissimilar frame works – RTC and EPICS, and there are a variety of languages used – C++, C, java, scripting languages. However, the interfaces can be placed into three groups, as described in the following three paragraphs.

As well as having a native interface called channel access, the EPICS systems in use at WMKO have a legacy communications layer (KTL) based on keywords for both command and telemetry. Tcl/Tk has been extended to interface to KTL keywords. This layer is well engineered and flexible, allowing easy communication across operating systems and between CPUs, so we wished to continue to use it in the interferometer EPICS systems. Thus, all the interfaces between EPICS systems in the interferometer, and between interferometer EPICS systems and telescope EPICS systems are implemented in the KTL frame work.

The RTC sub-systems at the real time (VxWorks) level (e.g. delay lines, fringe tracker) have a built in inter-processor communications protocol (IPC) that is used, for example, for communication of fringe tracker targets to the delay lines. IPC works between CPUs on the same VME back plane, and between CPUs in separate VME crates provided the physical distance is not large (this is important in the interferometer, where the CPUs for the interferometer are far from those controlling telescope systems).

For communication between the non-real time parts of RTC (e.g. GUIs, configuration data base), and from these parts to the real time sub-systems, a communications layer based on CORBA is provided. CORBA is inherently object oriented, and so fits well in the RTC design philosophy. CORBA and its application to RTC is beyond the scope of this article (see Lockhart[5] for more information), but CORBA is designed to make communication interfaces more transparent by confining user accessible parts of the interface to a C++ like definition called IDL. CORBA ORBs are available for all the languages (C++, Java, Tcl, Python) and all the operating systems (VxWorks, Unix/Linux) in use in the interferometer control system. We were thus able to construct a common interface based on IDL definitions for all RTC based systems. For example, one definition of telemetry exists that is used by all RTC based systems to report collected science data, systems health and system status. Use of CORBA thus makes interfacing between heterogeneous systems more straightforward and helps solve the interface complexity issues for the RTC based systems.

Thus, controlling interface complexity in the interferometer is largely reduced to providing a simple interface between CORBA and keyword based systems. Fortunately, much of this interface can also be confined to the sequencer level,

and specifically most communication can be conveniently routed through the interferometer sequencer without loss of performance or introduction of data flow complexity. We have thus provided both a CORBA based interface layer and a keyword based interface layer as parts of the Rhapsody frame work used to construct the interferometer sequencer. Thus, for example, this sequencer can command RTC systems based on CORBA method calls, and at the same time send keyword writes containing target information to telescope sequencers. No direct translation of keyword to CORBA systems is provided as the information needs in all cases to be processed through the sequencer, for it to provide its sequencing functionality.

There are a very few cases where it does not make sense to route commands and data passing between RTC and EPICS based systems through the sequencer layer. An example is the mirror position offload commands generated by the angle tracker RTC sub-system being sent to the EPICS based AO sub-system. In all these cases the interface is very simple, requiring only that a simple position, or position pair be passed (in the form of a primitive integer or float type) from CORBA to KTL or vice versa. In these cases we have provided a simple thin translation layer to implement the communication. These layers are implemented on UNIX to avoid complications of scheduling in VxWorks, and are generic enough to handle the simple interface requirements of all these special cases.

7. STATUS

At the present time the control system for the Keck Interferometer has been developed to the extent needed to perform traditional visibility science measurements using the two Keck telescopes. The following sub-systems have been implemented: two delay lines (one for each beam train), two angle trackers, two tip-tilt metrology systems, and one fringe tracker. In addition, sequencing of these sub-systems has been implemented and this sequencer interfaces to the telescope systems through a telescope sequencer to provide target information. Constant term metrology is used to measure the total path length variations along the whole beam train, and accelerometers are used to do likewise for the telescope optics. All optical elements that are routinely moved during set up, alignment, or calibrations (e.g. mirrors, shutters, retro-reflectors, etc) have been automated and are controlled by either the interferometer or auto-alignment sequencers, such that the interferometer can be run entirely remotely during normal operation. Extensive user interfaces have been provided in java and Tcl/Tk that also allow easy remote operation. Indeed the full interferometer has often been operated in internal calibration mode from JPL during day time testing.

Near term developments include: automated control of the Long Delay Lines to provide larger amounts of (fixed) optical path differencing to allow extended sky coverage, and inclusion of these into the auto-alignment system; development of the nulling beam combiner; and development of the differential phase mode of the instrument.

ACKNOWLEDGEMENTS

The work reported here was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration, and at the W. M. Keck Observatory, California Association for Research in Astronomy. Funding for the Keck Interferometer is provided by the National Aeronautics and Space Administration.

REFERENCES

1. See http://huey.jpl.nasa.gov/keck/, http://planetquest.jpl.nasa.gov/, http://origins.jpl.nasa.gov/
2. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer: progress report", in Interferometry in Optical Astronomy, Proc. SPIE, 4006, 2000.
3. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer update", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
4. See http://mesa53.lanl.gov/lansce8/EPICS
5. T. Lockhart, "RTC: a distributed real-time control system toolkit", Proc. SPIE, 4848, 2002 (this conference).
6. L. Reder, et al., "Using scripting languages in optical interferometry", Proc. SPIE, 4848, 2002 (this conference).
7. M. M. Colavita, et al., "The Palomar testbed interferometer", Ap.J., **510**, pp. 505-521, 1999

8. G. Vasisht et al., "Performance and verification of the Keck interferometer fringe detection and tracking system: FATCAT", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
9. See http://isc.caltech.edu/KISupport.
10. See http://www.ilogix.com.
11. G.T. Van Belle, et al., "Keck interferometer autoaligner", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).

Jet Propulsion Laboratory

4800 Oak Grove Drive, Pasadena, CA 91109-8099
(818) 354-5960 Fax (818) 393-1565

fax transmittal

to: ALEX LEVI / SANDRA DAWSON

fax number: 3-6734

from: DAVE FULTON

date: 9/9/02

subject: ARTICLE ON KECK-OUTRIGGERS

pages: 9

remarks:

- Thanks - Dave (X4-3262)

Overview of the control system for the Keck Interferometer

A. J. Booth[*a], G. Eychaner*[a], E. Hovland*[a], R. L. Johnson Jr.*[a], W. Lupton**[b], A. Niessner*[a], D. Palmer*[a], L. J. Reder*[a], A. C. Rudeen**[b], R. F Smythe*[a], K. Tsubota**[b]
[a]Jet Propulsion Laboratory, California Institute of Technology, [b]W. M. Keck Observatory, California Association for Research in Astronomy

ABSTRACT

The Keck Interferometer links the two 10m Keck Telescopes located atop Mauna Kea in Hawaii. It is the first 10m class, fully AO equipped interferometer to enter operation. Further, it is the first large interferometer designed to be handed over from a design and implementation team to a separate operations team, and be used by astronomers who are not interferometer specialists. As such it offers unique challenges in reducing an extremely complex and powerful system to an apparently simple user interface, and providing a well engineered system that can be maintained by people who did not develop it.

This paper gives an overview of the control system that has been implemented for the single baseline operation of the instrument, and indicates how this will be extended to allow control of the future modes of the instrument (nulling, differential phase and astrometry).

The control system has several parts. One is for control of "slow" sub-systems, which is based in the EPICS architecture, already ubiquitous at the Keck Observatory. Another, used to control hard real time sub-systems, is based on a new infrastructure developed at JPL, programmed in C++, Java, and using CORBA for communication. This infrastructure has been developed specifically with the problems of interferometric control in mind and is used in JPL's flight testbeds as well as the Keck Interferometer. Finally, a user interface and high level control layer is in development using a variety of tools including UML based modeling in the Rhapsody tool (using C++ and CORBA), Java, and Tcl/Tk for prototyping.

Keywords: Interferometry, control system, real time control

1. INTRODUCTION

1.1. Overview

The Keck Interferometric Array, as proposed, would use Michelson combination among the two Keck 10m telescopes along with four proposed 1.8m outrigger telescopes. The two Kecks provide a baseline of 85 m in the NE direction. With the addition of the outriggers, a range of baselines from 30 m to 135 m could be provided. For highest efficiency, all of the telescopes are phased: the Kecks use adaptive optics, while the outriggers would use fast tip/tilt correction. Cophasing among baselines is provided using active fringe tracking and active delay lines; for off-source cophasing, a dual-star module will be installed at each telescope to bring the light from the source and a cophasing reference to the beam combining lab. The back-end science instruments of the interferometer include two-way combiners at 1.5-2.4 μm for astrometry, traditional visibility science, and cophasing; a proposed multi-way imaging combiner at 1.5-5 μm; and a nulling combiner at 10 μm in development. The Keck Interferometer is the first 10m class, fully AO equipped interferometer to enter operation and is described in detail by Colavita and Wizinowich[2,3]. The Keck Interferometer is an element in NASA's Origins Program[1].

* Andrew.J.Both@jpl.nasa.gov; phone 1 818 354 1200; fax 1 818 393 4357; Jet Propulsion Laboratory, MS171-113, 4800 Oak Grove Drive, Pasadena, CA 91109; **ktsubota@keck.hawaii.edu; phone 1 808 885 7887; fax 1 808 885 4464; W. M. Keck Observatory, 65-1120 Mamalahoa Highway, Kamuela, HI 96743

The science goals of the interferometer are the use of nulling interferometry to characterize zodiacal dust emission from nearby solar type stars as a precurser to the SIM and TPF planet finder missions (nulling mode); to detect hot Jupiters (Jovian sized planets in close orbit around their parent stars) using differential phase signals at varying detector wavelengths (DP mode); astrometric searches for planets down to Uranus mass around nearby stars (astrometry mode); and general purpose high resolution astronomical imaging of a variety of sources using the full array (imaging mode). For a detailed description of the science goals of the instrument see Colavita and Wizinowich[2].

1.2. Control system

The control system for the Keck Interferometer is large and complicated. There are many parts and many interfaces, and it is being developed by two physically distant groups – JPL and WMKO, so there is ample potential for development of a system that is difficult to use and maintain. In addition, the system must be handed over from the development group (which for the interferometer itself, as opposed to telescope systems, is dominated by JPL personnel) to the operations group, which will contain none of the JPL developers. Further, the instrument must be of facility class, such that it may be used by astronomers who are not interferometry specialists. Thus, it is requirement that the instrument be, in fact, easy to use and maintain. In addition there are very tight requirements on the performance of the system to reach the challenging science goals of, for example, nulling detection of exo-zodiacal emission and differential phase measurements of hot Jupiters. These requirements are not easy to achieve.

The main components of the control system are:

- **Angle trackers,** which keep the light from the telescopes falling on the entrance aperture to the fringe trackers, and keep the wavefronts that are being combined to form interference fringes parallel.
- **Delay lines,** which keep the optical path difference between any two telescope beams that are being combined the same to within very tight tolerances whilst tracking the moving delay cause by the sidereal motion of the target object.
- **Metrology,** which monitors and corrects for path length and tilt variations in the optical beam trains of the interferometer from the telescopes to the detectors
- **Fringe trackers,** which not only measure the fringe position and provide feed back to delay lines to maintain lock on the interference, but also provide the measurements for phase and visibility that form the science observations.
- **Nulling and imaging cameras,** which will measure the fringe quantities of interest to nulling and imaging science.
- **Telescope systems,** being the telescope and dome control themselves, and also target acquisition, adaptive optics (AO) control, and coudé beam train control (particularly for off axis beam trains used for the secondary star feed).

For most of these sub-systems, several copies of functionally identical, but differently configured versions of the sub-systems are used together. For example, in a full imaging mode deployment there would be twelve delay lines operating simultaneously. There are numerous interfaces between the different sub-systems leading to a complicated data flow in the instrument. For example, as mentioned above, the fringe trackers measure the position of the interference fringes formed by combining the light from two telescopes. This position is fed back to the delay lines so as to keep the center of the interference fringe packet on the beam combiner, to maintain the highest fringe visibility. As another example the angle trackers feed back to the AO systems to keep the target object aligned with the detectors.

The tolerances on the performance of the sub-systems are very demanding. Servos must run at fast rates, for example the delay lines internally servo their positions against a metrology source at 5kHz. The angle trackers and fringe trackers need to read out their camera hardware at close to the maximum allowable rate to provide enough bandwidth for fringe tracking control and angle tracking control to overcome the variability induced by the earth's atmosphere and the internal variations in the instrument (fringe tracker frames at up to 500Hz and angle tracker frames at up to 100Hz). The servo control must also be very precise. The fringe tracking must control the fringe position to much better than a wavelength of light (so to better than 100nm or so), even though the delay lines are moving over 10s of meters during an observation.

To successfully make science measurements with the instrument there must be a high degree of automation for the control system. This also naturally leads to a control system that is easy to use since the user need not have a detailed

knowledge of the internal workings of the subsystems to achieve a set of science observations. The main reason for this automation requirement is that interferometry science requires high levels of careful calibration, thus we require:

- Fast switching between taking science data and taking internal calibration data.
- Fast switching between science target objects and objects of known visibility and phase for external fringe calibration.
- Consistency of calibration processes from targets to calibrator objects and in set up of target and calibrators.

These requirements demand automation at several levels:

- Automated sequencing of sub-system operation within a subsystem.
- Automated sequencing of sub-systems for coordinated operation of subsystems.
- Automated sequencing of targets presented to the instrument.
- Automated optical alignments of system and sub-system optical elements

This last automation requirement arises because the interferometer has many optical elements in each beam train from telescope to detector (many 10s of elements), and the total optical path is very long (many 10s of meters). Thus, manual alignment would be too time consuming and also not of sufficient accuracy.

2. DESIGN APPROACH

2.1. Software

The design approach taken for the interferometer control system was driven by the above complexities and requirements. In this section we give an overview of the thinking behind the current and planned implementations.

At WMKO extensive use is already made of the EPICS[4] control framework. Many observatory sub-systems, including telescope control, are written using EPICS, and there are additional tool sets developed by WMKO which are layered on top of EPICS, for example a keyword interface layer (KTL). There thus exists a large body of expertise and knowledge at WMKO in the use of EPICS and many control systems are built on thoroughly tested device control modules built in EPICS. It was therefore decided wherever possible, to make use of existing EPICS developments at WMKO for control of interferometer sub-systems, and to use WMKO expertise to extend existing control systems where necessary to control interferometer sub-systems that are similar to these existing control systems. This not only builds on existing infrastructure, but also helps the handover process as WMKO personnel are already familiar with the details of control systems for these interferometer elements, even if they were developed by JPL personnel. Care was taken to also make the development and deployment environment of EPICS systems at JPL adhere to WMKO policy and procedures.

Much of the interferometer's control system must run at rates too high to be implemented using EPICS. For these parts of the system we chose to implement control using the JPL developed RTC tool set[5]. The use of this tool set is described more fully below, however, here we note that the tool set allows a modular approach to design of the control system. We have many subsystems that are very similar in concept (for example, angle trackers and fringe trackers both read out a camera at high rate, compute a target, and send the target to another subsystem for actuation); we have many copies of the same type of subsystem, as noted above for example for the 12 delay lines needed in imaging mode. The object oriented nature of the RTC tool set allows easy modularisation of the elements of the control system to achieve this repeatability in design and in instantiation.

In addition to being implemented as a modular system, the control system is also very hierarchical. This particularly applies to the sequencing type of control for the system and for the sub-systems. Sequencing occurs at all levels within in the control system: At the lowest level, the sub-systems sequence their internal states based on low level input from sensors. For example, is there light in the detector, then start to track the fringes. At an intermediate level, sub-systems must be coordinated to act in concert. For example the observational target must be acquired by the angle trackers before the fringe trackers can try to track fringes. Also, the control system must cycle through various states of the sub-systems to achieve an observing sequence with the required internal calibrations. Finally at the highest level the control system must cycle through a predefined list of targets and calibration sources to achieve scientifically interesting measurements. The hierarchical nature of this sequencing allows us to implement it in a modular and hierarchical way, with separate control objects for each level of the hierarchy, linked by common format interfaces. This allows for

separate development and testing, and implementation of prototypes for various levels along side more fully developed systems at other levels.

For sub-systems developed within the EPICS framework, we also employed hierarchical development, layering sequencers on top of collections of individual sub-system modules, to implement, for example, the auto-alignment system.

We made extensive use of prototyping and iterative development of sub-systems and sequencers. Sub-systems are developed as stand alone modules, then interfaced to other subsystems and sequencers. They are also extended or redesigned when further development of the instrument is instigated, for example, development of the precision modes of nulling and differential phase from basic control achieved fro visibility science. Sequencing almost always was implemented first in scripting languages, allowing rapid prototyping and development[6].

2.2. Hardware

The above approach to software development: capitalizing on WMKO experience and making systems modular for ease of development and maintenance was also repeated for hardware. As far as possible we developed the control systems using hardware already in use at WMKO, or already in their planned upgrade path. We also repeated use of hardware between EPICS and RTC systems. This was fortunately aided by the fact that similar hardware and development environments to those previously used by WMKO were already in use at JPL for test bed development of the RTC tool kit. Thus, all our subsystems are deployed as Power PC control in a VME environment using the VxWorks operating system. We make extensive use of Industry Pack modules for D to A and A to D, and for digital IO, for example, and have reused and developed WMKO systems based on PMAC intelligent controllers. Many of the physical actuators used in the interferometer are identical to those already in use at WMKO, Newport 850G positioners, for example.

3. SUB-SYSTEM CONTROL

3.1. RTC controlled "fast" sub-systems

All the hard real time sub-systems in the interferometer control system that must run at "fast" rates (i.e. more than about 100Hz) are built using the JPL developed RTC control system[5]. This control system development tool has been developed at JPL as part of the interferometer development and test bed program, of which the Keck interferometer project is part. Thus, the framework has been developed with interferometer control especially in mind. The framework has been developed using C++, and Java, in both the VxWorks real time OS, and UNIX and Linux OS regimes. The framework is inherently object oriented.

The framework contains the following parts:

- Real time control system frameworks in C++ for VxWorks. The frameworks include real time state machines, device drivers, and predefined interfaces for commanding and telemetry.
- Servers for distributing telemetry in C++ under UNIX/Linux and an archiving system for capturing telemetry to disk, also in C++ for UNIX/Linux.
- An extensive relational data base system for configuration management, including a java based GUI, used to maintain configurable parameters for the real time systems making for easy deployment of multiple copies of sub-system controllers with differently configured properties. Reconfiguration can occur at run time, and even "on the fly".
- A flexible engineering GUI development tool kit developed in Java for UNIX/Linux/Windows, allowing on the fly configurable textual and graphical display of telemetry, and easy implementation of graphical command interfaces.
- Timing and scheduling control mechanisms and hardware for the real time processes.

In the following sub-sections we give some details of the real time sub-systems developed for the Keck Interferometer using this framework. Each description is for a single instance of each sub-system, of which there are generally several

in the interferometer system. All sub-systems are controlled independently by a single instance of their controlling software system, loaded with parameters from the configuration data base.

3.2. Delay lines

The delay lines are used to provide variable optical path length in each beam train from telescope to detectors, in response to sidereal motion of the target objects, and to provide an actuator for the fringe tracker system. They are implemented as a 4 stage nested servo system (similar to the delay lines used at the Palomar Testbed Interferometer[7]): an inner loop controlling a fast PZT mounted small mirror, two intermediate loops controlling voice coils actuating entire optics assemblies, and an outer loop controlling a micro-stepper motor that drives the whole assembly along rails. The inner loop responds to feed back error signals measuring the actual position of the assembly along its rail path using laser metrology, with a bandwidth of about 500Hz. Outer loops respond to this error at progressively lower bandwidths. Various targets are provided for the position of the assembly with regard to the laser metrology: a sidereal target from a higher level sequencer; a "continuous term" metrology signal, measuring the total path length of the beam train of which this delay line forms a part, from the fringe tracker to the telescope; and a measurement of the change in optical path length caused by motion of the telescope optics, measured using accelerometers mounted on the telescope. The closed loop target from the fringe tracker is either an actual error signal from a measurement of fringe position, or a search target if the tracker detects no fringes.

In common with all the real time control sub-systems, the RTC framework has been used to implement a state machine that allows the delay lines to be either "idle" (measuring sensor input, but not moving closed loop), or tracking. In tracking, the sub-system can be in a variety of states, in this case: slewing (moving rapidly to a new target), or locked (closed loop tracking on error signal). The RTC framework allows easy implementation of the state machine that defines these states and controls transition between them based on user input and sensor values.

3.3. Fringe trackers

The fringe tracker is implemented as two separate processes running on the same CPU. One process controls the camera readout, and one performs the servo calculations. The camera readout process is very simple, loading the appropriate clock pattern to read out the array and setting up the rate at which the pattern is applied, then placing the results of the readout into local memory. The servo process obtains the read out pixels from the camera, and sends a calculated target to the delay line sub-systems to keep the fringe packet centered on the beam combiner. The calculation involves determining a fringe phase from a white light pixel from one output of the beam combiner, and a fringe position group delay determined from a dispersed spectrum from the other output, then combining these two. Details of this process can be found in Vashist[8]. The state machine for the fringe tracker allows for idle and tracking states. In tracking the process can be searching (sending a linear spiral search target to the delay lines instead of an error signal), semi-lock (confirming the existence of suspected fringes), or lock (confirmed detection of fringes, and servoing delay lines).

3.4. Angle trackers

The angle trackers also have a camera readout process and a servo process. The camera readout process is virtually identical to that of the fringe tracker. The servo process determines a position for the target image on the camera and calculates an error signal to be sent to the tip-tilt metrology sub-system to keep the target centered on the detector. The target is determined using either a centroid algorithm, in initial wide angle mode, or a quad cell algorithm once the target has been acquired. The state machine allows for idle and tracking states, and for search, semi-lock and lock in tracking. In addition to providing an error signal to the tip-tilt system, the angle tracker also offloads the absolute tip-tilt mirror position to the telescope tracking systems (in the case of the Keck telescopes, this means the AO system) at a lower rate than the main servo to keep the mirrors close to center.

3.5. Tip-tilt metrology

The tip-tilt metrology sub-system performs a higher bandwidth control of the internal beam train tip-tilt alignment than can be achieved by the angle tracker. The angle tracker works using detected starlight from the telescope, so is limited by the photon rate of the object. The tip-tilt sub-system uses a laser beacon near the beam combiner at the fringe tracker to illuminate a detector near the telescope at the other end of the beam train. Tip and tilt are calculated from the laser

spot position on the detector, and fed back to a control mirror to keep the spot immobile. Targets are also added in from the angle trackers as mentioned above to keep the star centered on its detector (and hence ensure that light is correctly falling on the fringe trackers). The state machine for this sub-system uses the idle and track states. In the track state the state machine is very simple, allowing for a lock state if there is laser light falling on the detector.

4. SEQUENCING

The sub-systems described above run in hard real time, controlled by hardware clocks and with scheduling that requires the tasks complete in less time than required by the servo rate. This ensures optimum performance of the closed loop servos that are the primary concern of these sub-systems. These systems interface to a higher level of control that does not have to be run in hard real time, and thus has not been implemented in the RTC frame work. We refer to this level of control as sequencing, and it satisfies several requirements for the interferometer control system as indicated above: making the various subsystems act in concert, controlling internal calibrations for a single target object, and sequencing through a set of science and calibrator target objects.

As indicated above the sequencing systems are implemented as a hierarchical structure. Fig. 1 shows the hierarchy of the sequencing components. The subsystem sequencers are instantiated as one per real time sub-system, and control their individual subsystem via commands and monitored telemetry. They ensure that the more generic commands received from the interferometer sequencer are correctly interpreted in the command set of the real time sub-systems, and that those systems enter the expected state as a consequence of the commands. They allow for faulted states if the sub-systems do not perform in the expected manner. The interferometer sequencer contains several modules that perform the correct coordination of the real time subsystems (and the telescope systems) to achieve a science observation, including sequencing through internal calibrations. These internal calibrations include commands to "slow" subsystems implemented in EPICS (e.g. commands to close shutters) as well as "fast" systems implemented in RTC.

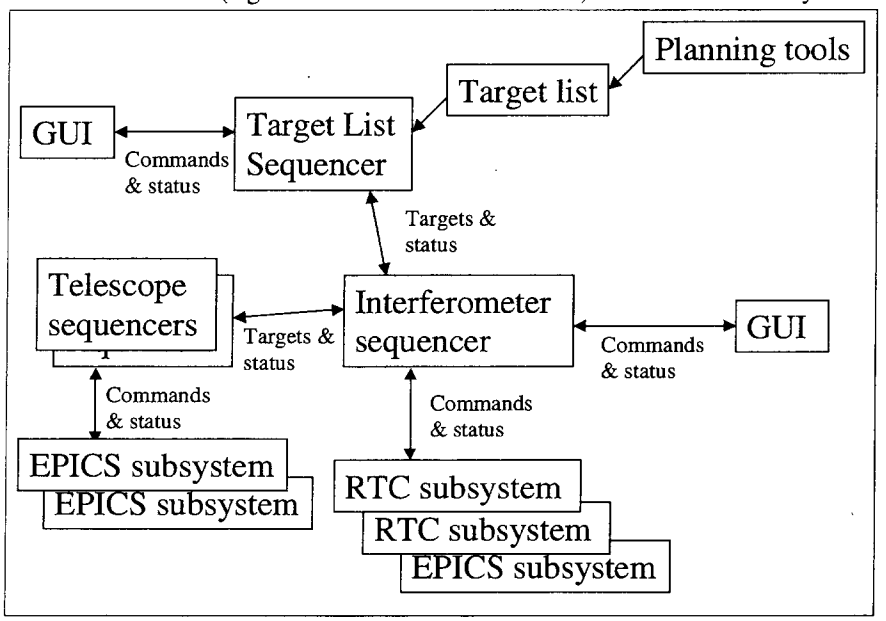

Fig. 1 Sequencing hierarchy

At the highest level the target list sequencer controls the interferometer sequencer and performs the function of cycling between science and calibration targets based on a target list supplied by planning tools[9].

The interferometer sequencer, sub-system sequencers and target list sequencer are being implemented using the "Rhapsody" tool from I-Logix[10]. This is a UML based tool with VPE interface which generates C++ code in an object oriented frame work. As mentioned, prototyping for sequencers is performed using scripting languages, and at present the target sequencer is implemented in Tcl/Tk. Telescope sequencers and telescope subsystem sequencers are being

implemented in the EPICS state notation language frame work. All these sequencers are currently implemented under the UNIX operating system.

5. AUTO-ALIGNMENT

As indicated above, an optical system as complicated as the Keck interferometer must have an auto-alignment system to allow quick reconfiguration, alignment checks and improvement during observations, improved accuracy and repeatability of alignment and improved human interface to optical alignment where the optics are located in a hostile environment (14000ft altitude). Details of the auto-alignment system are given by Van Belle, et al.[11].

The auto-alignment control system is implemented in a very similar manner to the main instrument control system. Individual sub-systems (e.g. 2 axis mirror controllers) are implemented in EPICS, often with self contained sequencers for automated start up, slew to position, etc. These systems are real time and run under the VxWorks operating system. At a higher level, there are sub-system sequencers, in this case implemented in the EPICS State Notation Language (SNL) frame work, controlling the external command and telemetry interface. Above these is a further layer of sequencing that controls related subsystems to achieve auto-alignment on a single optic, again in EPICS SNL. A typical sequence involves turning on a stimulus, detecting a centroid for a target on a camera, and servoing the optic position to center the target. At the highest level, a sequencer is provided to step through all the optical elements in a beam train (or all those that need to be touched after, say, a reconfiguration) and perform an alignment. At present this top level sequencing is implemented as a prototype in Tcl/Tk, but will shortly be ported to Rhapsody, with a java GUI.

6. INTERFACE CONTROL

It can be seen from the foregoing that the total number of interfaces in the interferometer control system is large, and that they are potentially of a diverse nature. There is inter-communication between systems running under different operating systems – VxWorks and UNIX/Linux, and systems built using two very dissimilar frame works – RTC and EPICS, and there are a variety of languages used – C++, C, java, scripting languages. However, the interfaces can be placed into three groups, as described in the following three paragraphs.

As well as having a native interface called channel access, the EPICS systems in use at WMKO have a legacy communications layer (KTL) based on keywords for both command and telemetry. Tcl/Tk has been extended to interface to KTL keywords. This layer is well engineered and flexible, allowing easy communication across operating systems and between CPUs, so we wished to continue to use it in the interferometer EPICS systems. Thus, all the interfaces between EPICS systems in the interferometer, and between interferometer EPICS systems and telescope EPICS systems are implemented in the KTL frame work.

The RTC sub-systems at the real time (VxWorks) level (e.g. delay lines, fringe tracker) have a built in inter-processor communications protocol (IPC) that is used, for example, for communication of fringe tracker targets to the delay lines. IPC works between CPUs on the same VME back plane, and between CPUs in separate VME crates provided the physical distance is not large (this is important in the interferometer, where the CPUs for the interferometer are far from those controlling telescope systems).

For communication between the non-real time parts of RTC (e.g. GUIs, configuration data base), and from these parts to the real time sub-systems, a communications layer based on CORBA is provided. CORBA is inherently object oriented, and so fits well in the RTC design philosophy. CORBA and its application to RTC is beyond the scope of this article (see Lockhart[5] for more information), but CORBA is designed to make communication interfaces more transparent by confining user accessible parts of the interface to a C++ like definition called IDL. CORBA ORBs are available for all the languages (C++, Java, Tcl, Python) and all the operating systems (VxWorks, Unix/Linux) in use in the interferometer control system. We were thus able to construct a common interface based on IDL definitions for all RTC based systems. For example, one definition of telemetry exists that is used by all RTC based systems to report collected science data, systems health and system status. Use of CORBA thus makes interfacing between heterogeneous systems more straightforward and helps solve the interface complexity issues for the RTC based systems.

Thus, controlling interface complexity in the interferometer is largely reduced to providing a simple interface between CORBA and keyword based systems. Fortunately, much of this interface can also be confined to the sequencer level,

and specifically most communication can be conveniently routed through the interferometer sequencer without loss of performance or introduction of data flow complexity. We have thus provided both a CORBA based interface layer and a keyword based interface layer as parts of the Rhapsody frame work used to construct the interferometer sequencer. Thus, for example, this sequencer can command RTC systems based on CORBA method calls, and at the same time send keyword writes containing target information to telescope sequencers. No direct translation of keyword to CORBA systems is provided as the information needs in all cases to be processed through the sequencer, for it to provide its sequencing functionality.

There are a very few cases where it does not make sense to route commands and data passing between RTC and EPICS based systems through the sequencer layer. An example is the mirror position offload commands generated by the angle tracker RTC sub-system being sent to the EPICS based AO sub-system. In all these cases the interface is very simple, requiring only that a simple position, or position pair be passed (in the form of a primitive integer or float type) from CORBA to KTL or vice versa. In these cases we have provided a simple thin translation layer to implement the communication. These layers are implemented on UNIX to avoid complications of scheduling in VxWorks, and are generic enough to handle the simple interface requirements of all these special cases.

7. STATUS

At the present time the control system for the Keck Interferometer has been developed to the extent needed to perform traditional visibility science measurements using the two Keck telescopes. The following sub-systems have been implemented: two delay lines (one for each beam train), two angle trackers, two tip-tilt metrology systems, and one fringe tracker. In addition, sequencing of these sub-systems has been implemented and this sequencer interfaces to the telescope systems through a telescope sequencer to provide target information. Constant term metrology is used to measure the total path length variations along the whole beam train, and accelerometers are used to do likewise for the telescope optics. All optical elements that are routinely moved during set up, alignment, or calibrations (e.g. mirrors, shutters, retro-reflectors, etc) have been automated and are controlled by either the interferometer or auto-alignment sequencers, such that the interferometer can be run entirely remotely during normal operation. Extensive user interfaces have been provided in java and Tcl/Tk that also allow easy remote operation. Indeed the full interferometer has often been operated in internal calibration mode from JPL during day time testing.

Near term developments include: automated control of the Long Delay Lines to provide larger amounts of (fixed) optical path differencing to allow extended sky coverage, and inclusion of these into the auto-alignment system; development of the nulling beam combiner; and development of the differential phase mode of the instrument.

ACKNOWLEDGEMENTS

The work reported here was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration, and at the W. M. Keck Observatory, California Association for Research in Astronomy. Funding for the Keck Interferometer is provided by the National Aeronautics and Space Administration.

REFERENCES

1. See http://huey.jpl.nasa.gov/keck/, http://planetquest.jpl.nasa.gov/, http://origins.jpl.nasa.gov/
2. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer: progress report", in Interferometry in Optical Astronomy, Proc. SPIE, 4006, 2000.
3. M. M. Colavita, and P. L. Wizinowich, "Keck Interferometer update", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
4. See http://mesa53.lanl.gov/lansce8/EPICS
5. T. Lockhart, "RTC: a distributed real-time control system toolkit", Proc. SPIE, 4848, 2002 (this conference).
6. L. Reder, et al., "Using scripting languages in optical interferometry", Proc. SPIE, 4848, 2002 (this conference).
7. M. M. Colavita, et al., "The Palomar testbed interferometer", Ap.J., **510**, pp. 505-521, 1999

8. G. Vasisht et al., "Performance and verification of the Keck interferometer fringe detection and tracking system: FATCAT", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).
9. See http://isc.caltech.edu/KISupport.
10. See http://www.ilogix.com.
11. G.T. Van Belle, et al., "Keck interferometer autoaligner", in Interferometry in Optical Astronomy II, Proc. SPIE, 4838, 2002 (this conference series).